Early-career Preparation

Rex Ying

Assistant Professor of Computer Science

Yale University

We Want Our Research to Have Impact

- **Central question:**
  - How does your research benefit the **research community** and **the society**?

- Impact is a multi-dimensional goal
  - Publications, talks, conference presentations
  - Teaching, mentorship
  - Engineering, tool-building, benchmarks
  - Collaborations within and beyond CS

Research Career Preparation

- Creating Impact
- Engineering Practice
- Experience as an AP

Research Career Preparation

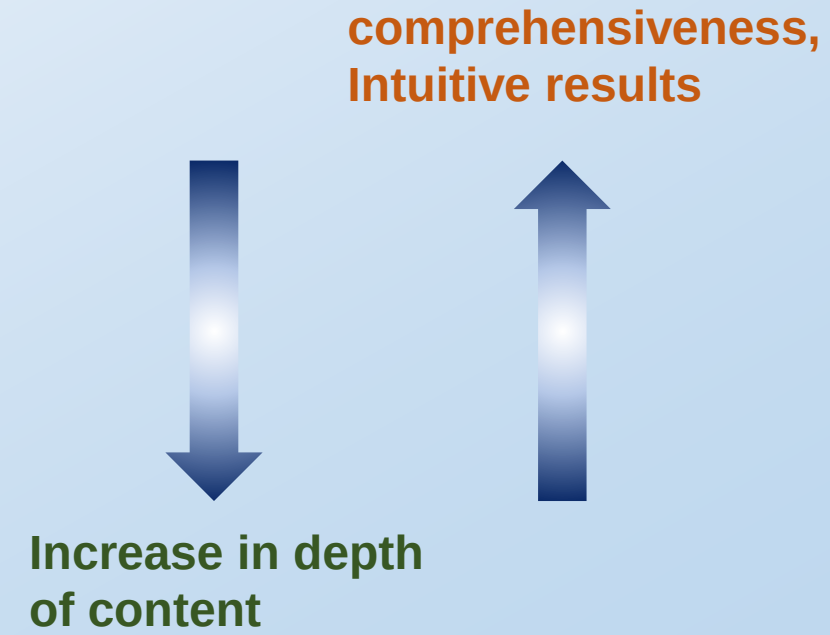
- **Creating Impact**
- Engineering Practice
- Experience as an AP

Publications

- Direct Impact through **Research papers and publication**
  - Idea, intuition, clarity through figures and text, experiments

- **Publication** is not the whole story

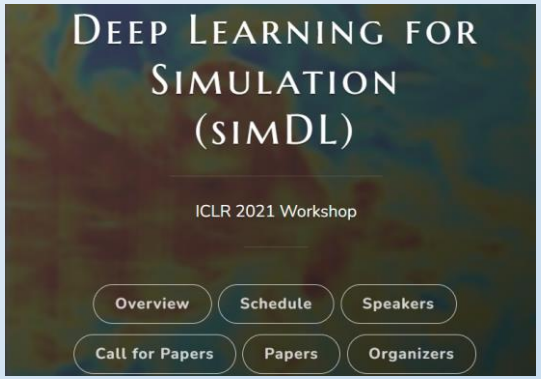
- Be aware of the types of audience
  - General public, students
  - Researchers / engineers with CS background
  - AI / ML practitioners and researchers
  - Experts in your field



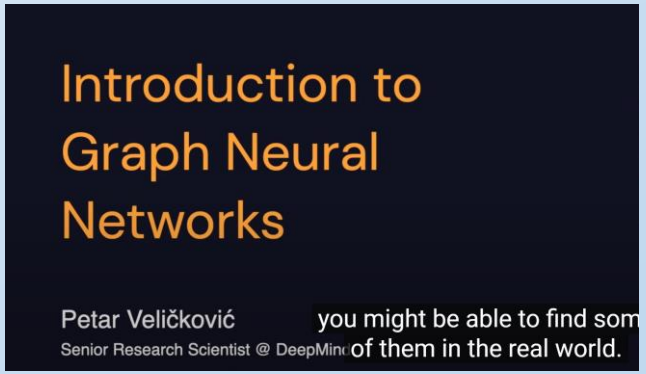
Presentations and Talks

- Talks are great ways to **promote your research**, share your findings and let people be more aware of its significance
- **Venues:**
  - Conferences, tutorials, workshops
  - Other academic institutions, industry collaborators, grant meetings
- Do not limit yourselves to only ML conferences!

Conferences



Workshops



Research talks

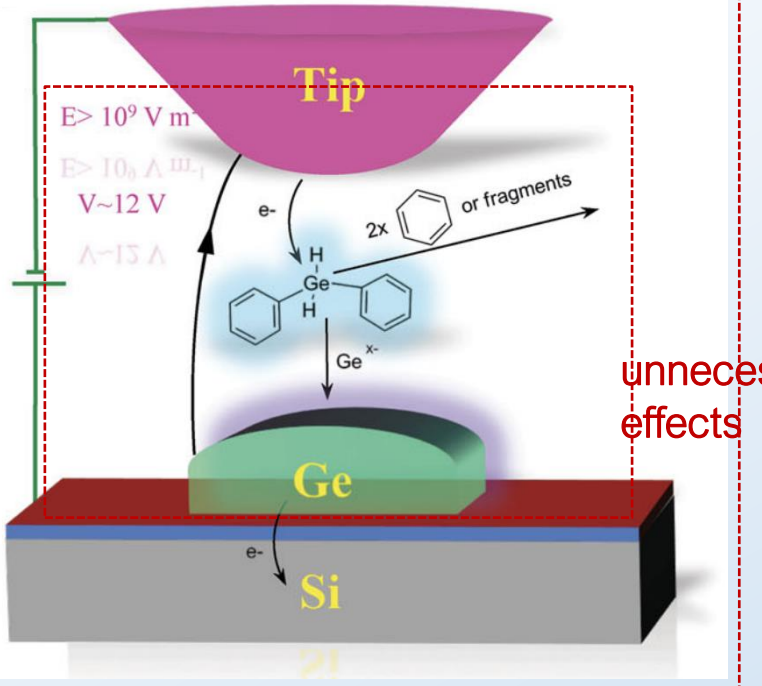
Writings and Figures

- It is incredibly important to ensure that
  - The writing is clear, rigorous
  - The figures are well-presented, illustrative

- Writing: Jennifer Widom's Guide
  https://cs.stanford.edu/people/widom/paper-writing.html

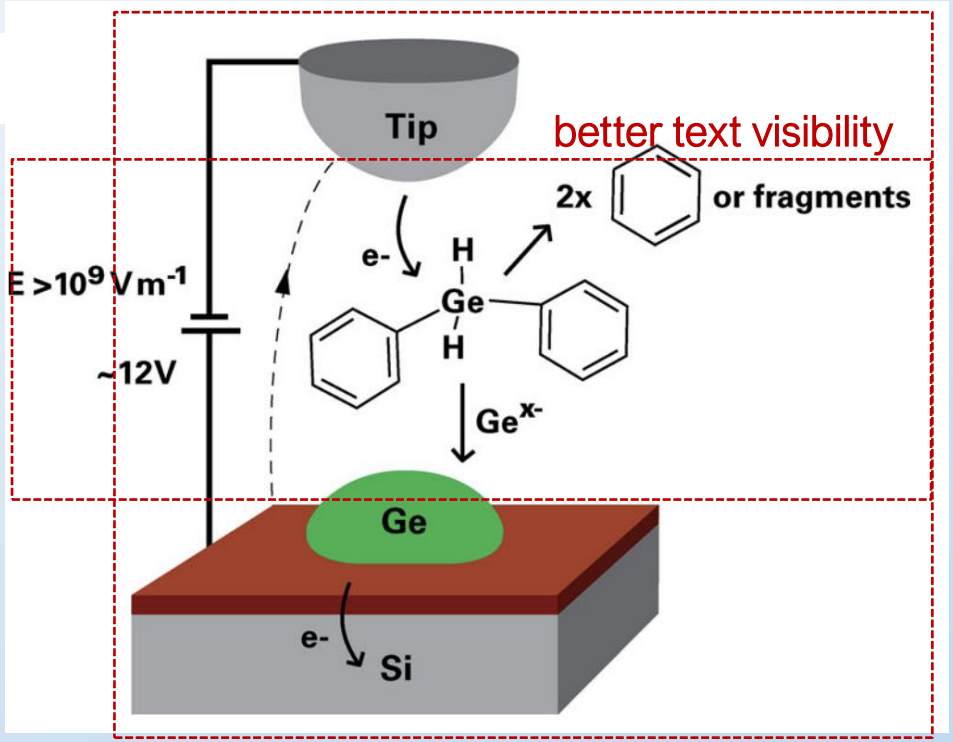
Version 1

too many colors



unnecessary effects

Version 2



better use of components and components are aligned on an implied grid

Credit: Maria Bribic

Rolandi, Cheng, Perez-Kriz: A Brief Guide to Designing Effective Figures for the Scientific Paper, *Advanced Materials* '11

- Try to make it memorable!
- Visualize key insights of your approach
- Figure follows abstract
- Catchy name of the method

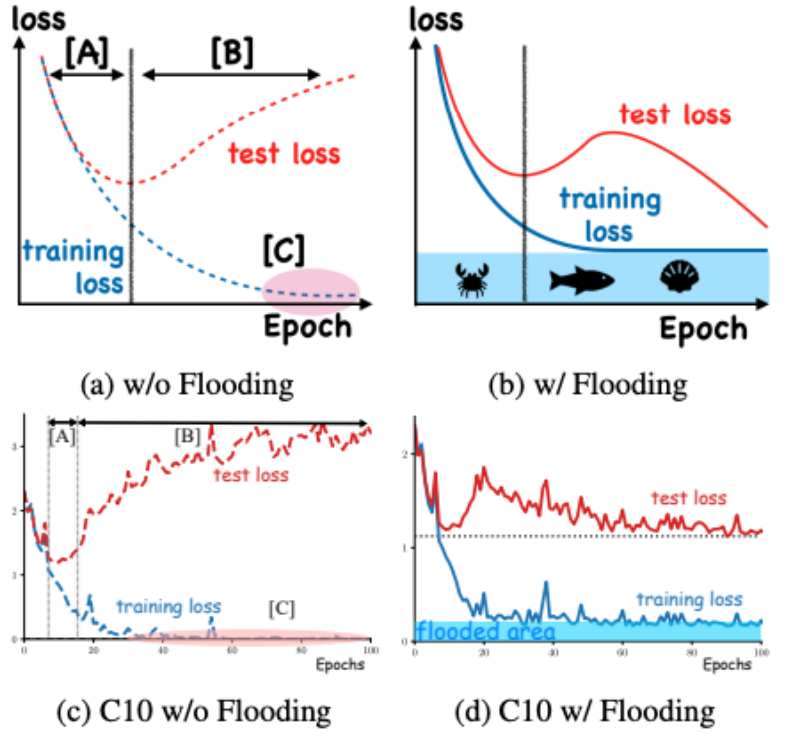
Approach is clear (& memorable) from the abstract and figure!

Do We Need Zero Training Loss After Achieving Zero Training Error?

Takashi Ishida [1 2] Ikko Yamane [1] Tomoya Sakai [3] Gang Niu [2] Masashi Sugiyama [2 1]

Abstract

Overparameterized deep networks have the capacity to memorize training data with zero *training error*. Even after memorization, the *training loss* continues to approach zero, making the model overconfident and the test performance degraded. Since existing regularizers do not directly aim to avoid zero training loss, it is hard to tune their hyperparameters in order to maintain a fixed/preset level of training loss. We propose a direct solution called *flooding* that intentionally prevents further reduction of the training loss when it reaches a reasonably small value, which we call the *flood level*. Our approach makes the loss float around the flood level by doing mini-batched gradient descent as usual but gradient ascent if the training loss is below the flood level. This can be implemented with one line of code and is compatible with any stochastic optimizer and other regularizers. With flooding, the model will continue to "random walk" with the same non-zero training loss, and we expect it to drift into an area with a flat loss landscape that leads to better generalization. We experimentally show that flooding improves performance and, as a byproduct, induces a double descent curve of the test loss.



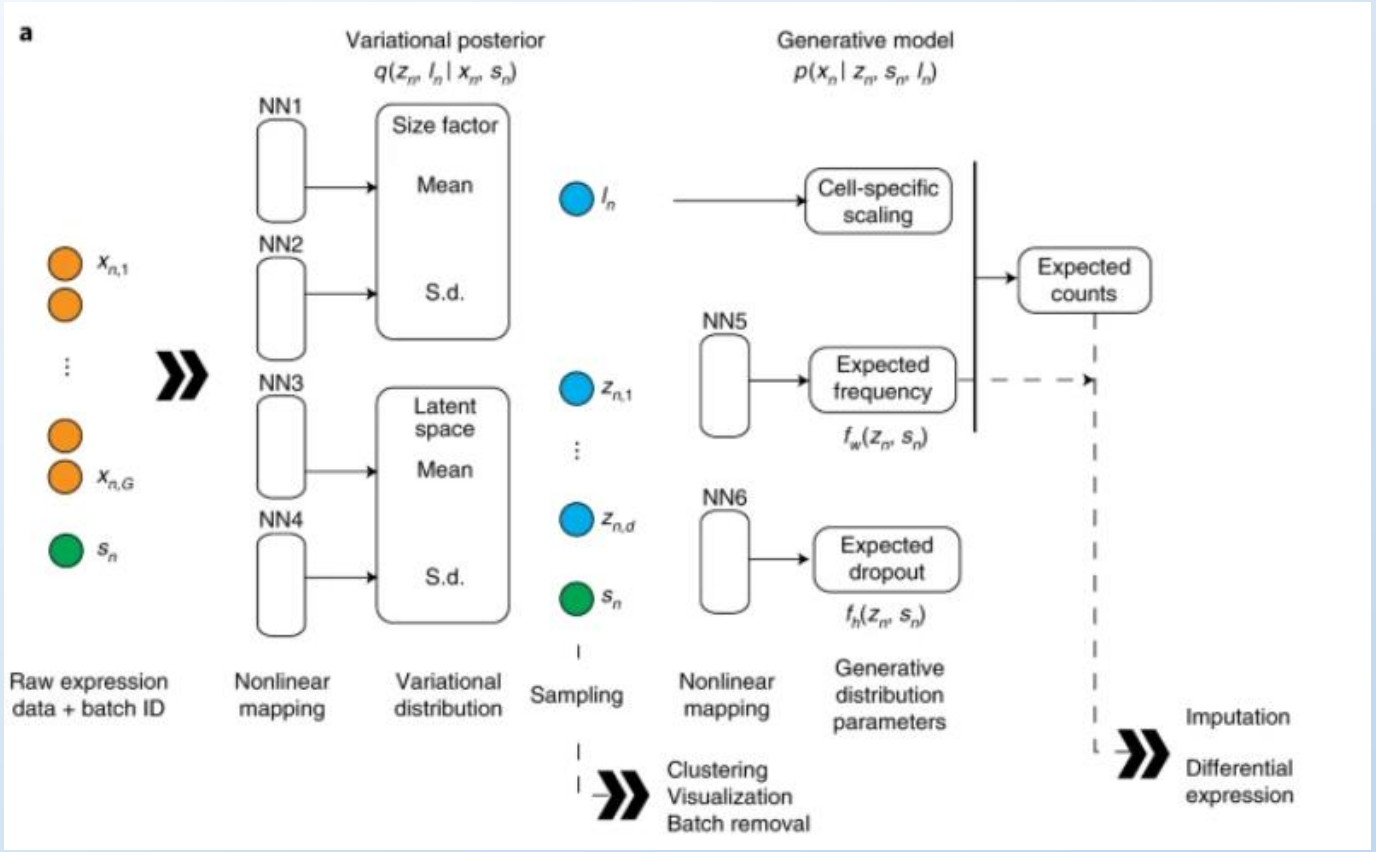
Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Credit: Maria Bribic

- Key insight/idea missing: What is unique about this approach? What is this method doing?
- Too many details
- Many symbols that are not introduced

  …
- Nothing memorable about it

What you do not want to do:



Credit: Maria Bribic

Teaching and Mentorship

- If you want to join academia, **learning to teach** is crucial
- Be a teaching assistant
  - Make pedagogical **slides** that are easy to understand
  - **Homework** / exam questions
  - **Office hours**: answer questions but do not directly give away answers
  - Students might be inspired by your research or apply your research in their future jobs!
- Be a research mentor (when you are more senior in research)
  - Take research assistantship / independent study students interested in research
- Organize **reading groups** on research topics

Impact: Tool-building

- Beyond talks, there are other opportunities to promote your research
- Be proactive in identifying such opportunities
- Tool building is a great way to accelerate research in the field
  - **Research codebase** with high coding standards
  - **Libraries** on a specific research field
  - **Benchmarks** and evaluation frameworks
  - These tools will also make it easy when applying your research to real-world use cases

Benchmarks and Evaluation Frameworks

- Examples
  - OGB: large scale graph learning benchmarks
  - GraphFramEx: graph explainability evaluation and benchmarks
  - DawnBench: training and inference speed / performance benchmarks
  - GraphWorld: synthetic graphs with diverse structure
  - Knowledge graphs, molecules, proteins, physical simulations, graph generative models …
- Again, be **proactive** in finding a **unique angle** to evaluation and benchmarks

Competitions

- Organize or participate in competitions

- KDD Cup Competitions
  - Graph AutoML
  - OGB large-scale challenge
  - Focus on a specific area / challenge that with real-world significance
- NeurIPS Competition Track

Collaborations

- Impact beyond machine learning / CS: **interdisciplinary research**
  - Many professors are willing to explore ML methods in their research
  - Talk to people from different backgrounds at the university, conferences or other venues
- **Industrial internships** are very important!
  - Understand real-world challenges
  - Get access to data that are very different from typical ML benchmarks
  - Create collaboration between the lab and the company
  - Get recommendation letter!
- **Grants** with other universities / industrial partners

PhD Fellowships

- There are many opportunities to obtain a **PhD fellowship**
  - They provide financial support
  - some may require / suggest internships
  - Google, Apple, Nvidia, Meta etc. all provide fellowship opportunities
    - Search online for eligibility. Be prepared when eligible.
  - Some of them may require **nomination** from the department
- **Benefits**
  - Great to appear in your CV
  - Establish bonds with industry

Constantly look for such opportunities

Summary

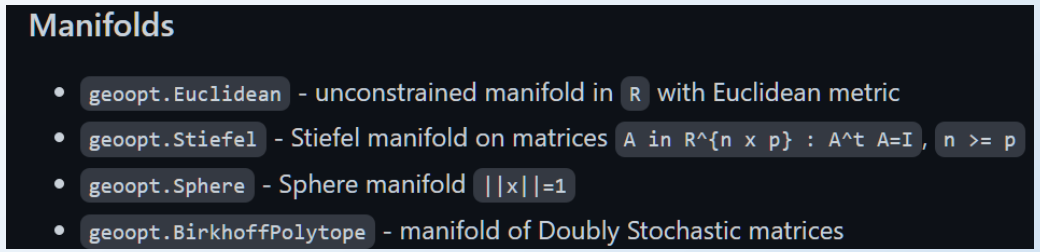
- Impact is a multi-dimensional goal
  - **Research**: publications, conference presentations, workshops, tutorials, talks
  - **Teaching**: TA, mentoring, reading groups
  - **Tool-building**: high-quality code, libraries, benchmarks, competitions
  - **Collaborations** (academia, industry, cross-disciplinary)
  - **Fellowships and awards**
- Communication is crucial
  - Talk to me when you need help
- Be **proactive**!!

Research Career Preparation

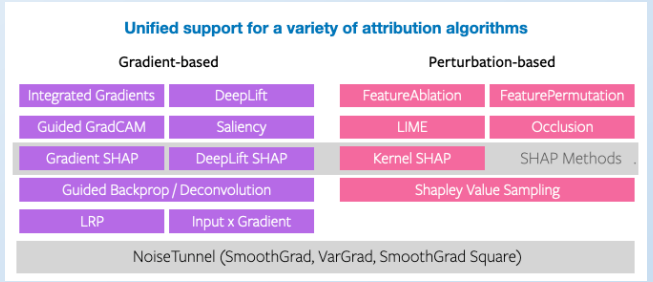
- Creating Impact
- **Engineering Practice**
- Experience as an AP

Libraries that Benefit Research

- As mentioned, **tool-building** (such as open-source libraries) is an integral part of research and impact creation
- Great opportunity to promote one's research
- Create a fair playground for new researchers
- Topics can be very broad or very specific



Geometry Library (GeoOpt)



Explainability Library (Captum)

AutoML Library (Ray Tune)

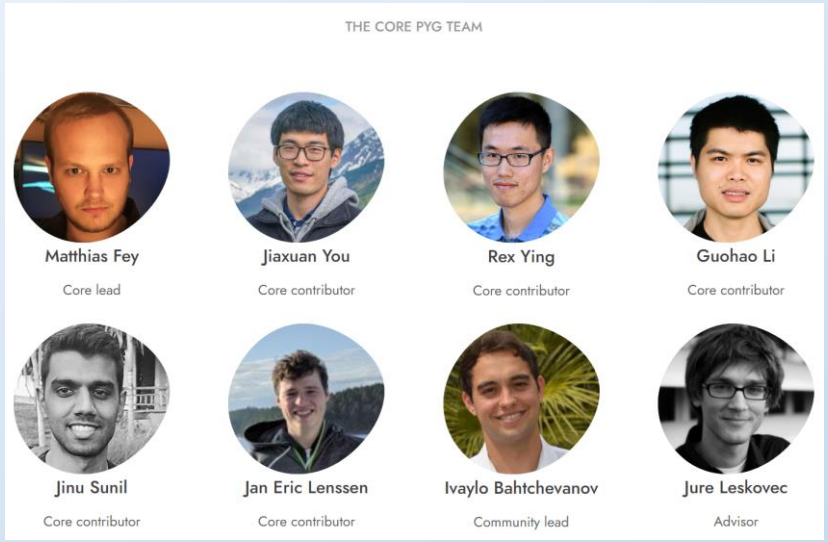
High Quality Codebase

- High-quality code is crucial for reproducibility in research
- Increase **impact of research** by allowing different audience to use your code
- **Principle**
  - **Simple API**! With very few lines of code, the user should be able to load in their own data, train a model, and make inference / prediction
  - **Modular**: make sure different functionalities are compartmentalized. Use OOP (Python allows multiple inheritance - Mixins!)
- More about coding and engineering practices in a separate session

This is typically a starting point for great libraries and tools!

Libraries

- Learning framework for a class of models: PyG / DGL

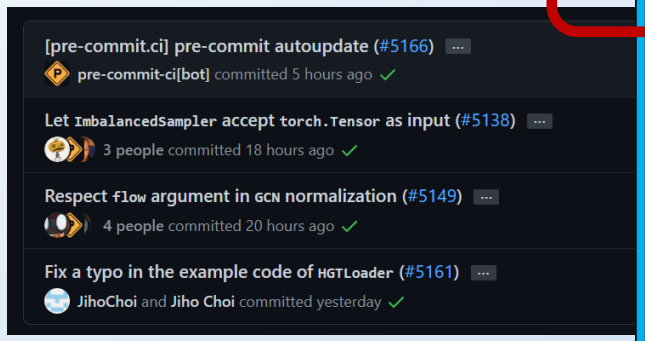


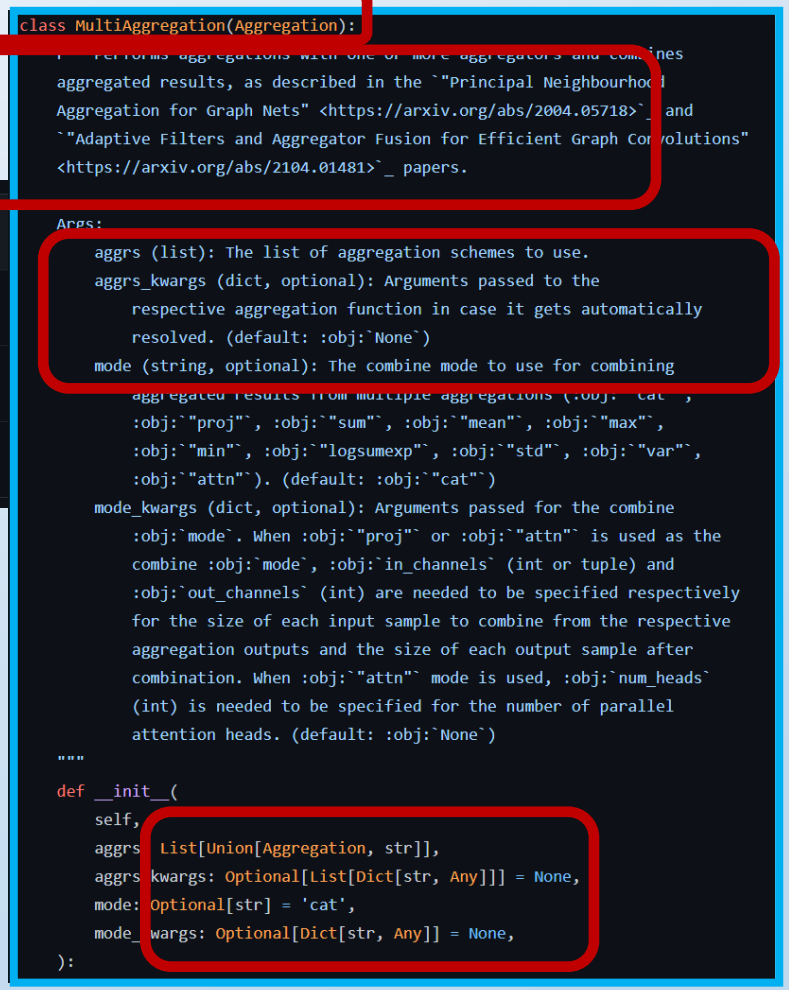
- Framework targeting use cases: KGE,
- Specific sub-topic: PyG-temporal, GNN AutoML
- There is high variation

APIs, Engineering designs, Efficiency, Coding styles and Active maintenance are all crucial factors!!

Engineering Practice

- Good coding styles
- Use versioning
  - Revert when there is a bug
- Use pull requests
- Github actions; unit tests
- Record experiment settings
  - Use AutoML / experiment management tools
- Keep **visualization code** and Jupyter notebooks
  - MLflow, Tensorboard
- Use documentation (docstring)!





Type Hints

- Make sure you use typing for function signatures

```
-     def __init__(self, edge_model=None, node_model=None, global_model=None):
93    +     def __init__(
94    +         self,
95    +         edge_model: Optional[torch.nn.Module] = None,
96    +         node_model: Optional[torch.nn.Module] = None,
97    +         global_model: Optional[torch.nn.Module] = None,
98    +     ):
```

- Consider careful use of Optional, Any, Union …
- Specify output

```
-     def forward(self, batch=None):
+     def forward(self, batch: OptTensor = None) -> Tensor:
```

Annotation

- Annotation can simplify code and provides readability

```python
@property
def has_data(self) -> bool:
    return getattr(self, '_data', None) is not None
```

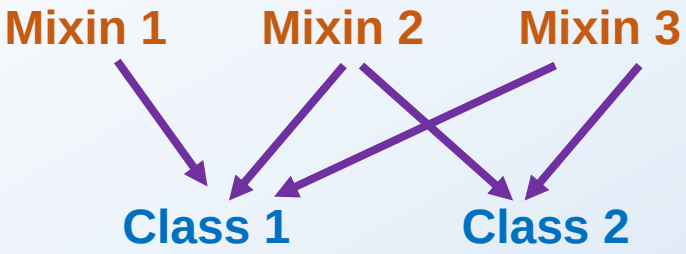
```python
@classmethod
def raise_post_materialization(cls, func):
```

- Custom-defined annotations

```python
@Timer()
def add_metapaths(self, metapaths: List[List[EdgeType]],
                  no_self_loops: bool = True,
                  max_sample: Optional[int] = None,
                  weighted: bool = False):
```

Object-oriented Design

- Python allows multiple inheritance

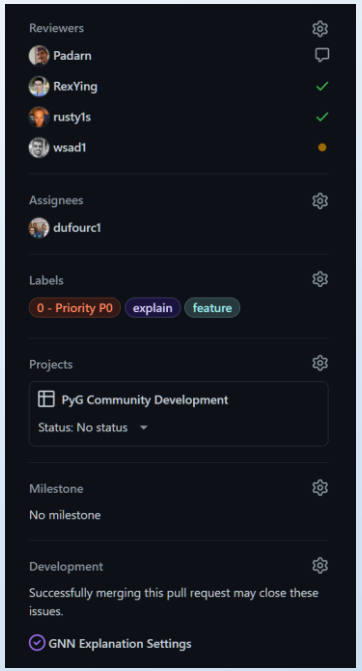


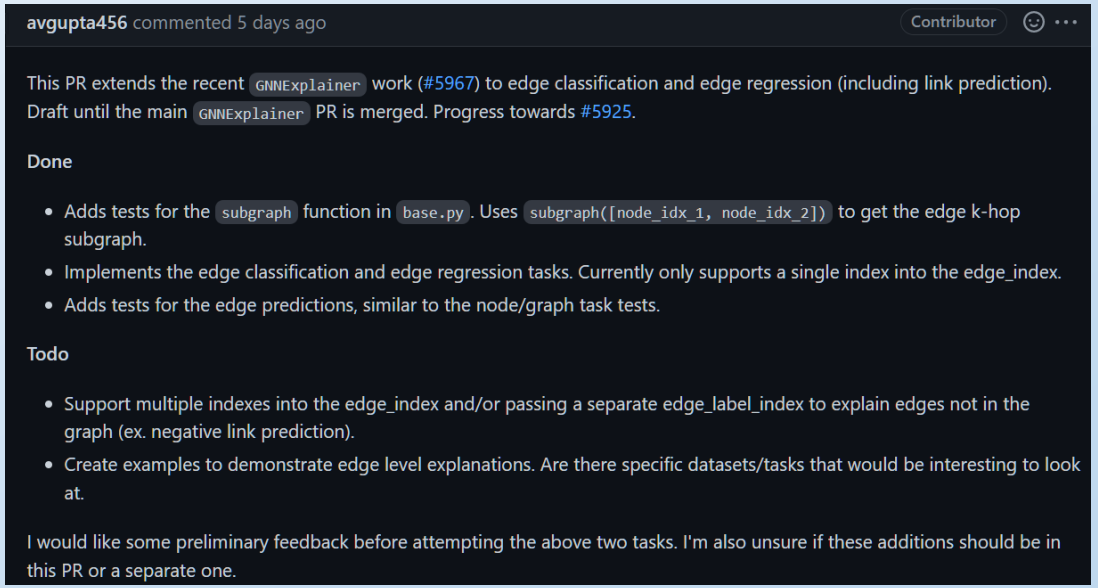

```python
class CastMixin:
    @classmethod
    def cast(cls, *args, **kwargs):  # TODO Can we apply this recursively?
        if len(args) == 1 and len(kwargs) == 0:
            elem = args[0]
            if elem is None:
                return None
            if isinstance(elem, CastMixin):
                return elem
            if isinstance(elem, (tuple, list)):
                return cls(*elem)
            if isinstance(elem, dict):
                return cls(**elem)
        return cls(*args, **kwargs)
```

- Each mixin tackles one aspect (e.g. configuration; type casting; training framework etc.)

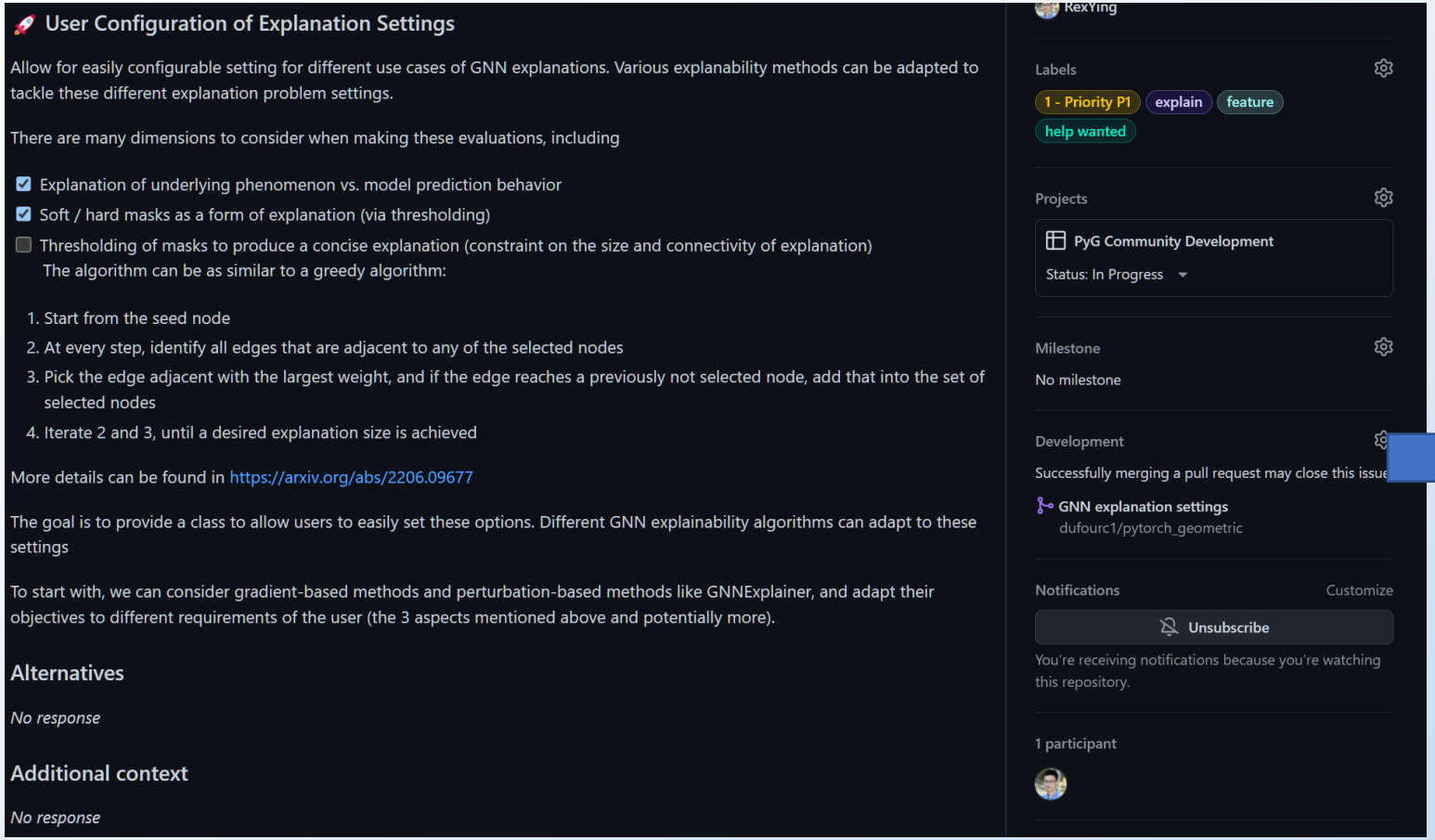
Pull Requests (PRs) for Collaborative Projects

- Description needs to be clear
  - Contexts are provided
  - Checklists, visualizations can help illustrate the utility
- Reviewers
- Assignees
- Labels
- Projects
- Associated issues





More Issues / PR Description Examples



🚀 User Configuration of Explanation Settings

Allow for easily configurable setting for different use cases of GNN explanations. Various explanability methods can be adapted to tackle these different explanation problem settings.

There are many dimensions to consider when making these evaluations, including

- [x] Explanation of underlying phenomenon vs. model prediction behavior
- [x] Soft / hard masks as a form of explanation (via thresholding)
- [ ] Thresholding of masks to produce a concise explanation (constraint on the size and connectivity of explanation)
  The algorithm can be as similar to a greedy algorithm:

1. Start from the seed node
2. At every step, identify all edges that are adjacent to any of the selected nodes
3. Pick the edge adjacent with the largest weight, and if the edge reaches a previously not selected node, add that into the set of selected nodes
4. Iterate 2 and 3, until a desired explanation size is achieved

More details can be found in https://arxiv.org/abs/2206.09677

The goal is to provide a class to allow users to easily set these options. Different GNN explainability algorithms can adapt to these settings

To start with, we can consider gradient-based methods and perturbation-based methods like GNNExplainer, and adapt their objectives to different requirements of the user (the 3 aspects mentioned above and potentially more).

Alternatives

No response

Additional context

No response

RexYing

Labels

1 - Priority P1 | explain | feature | help wanted

Projects

PyG Community Development

Status: In Progress

Milestone

No milestone

Development

Successfully merging a pull request may close this issue

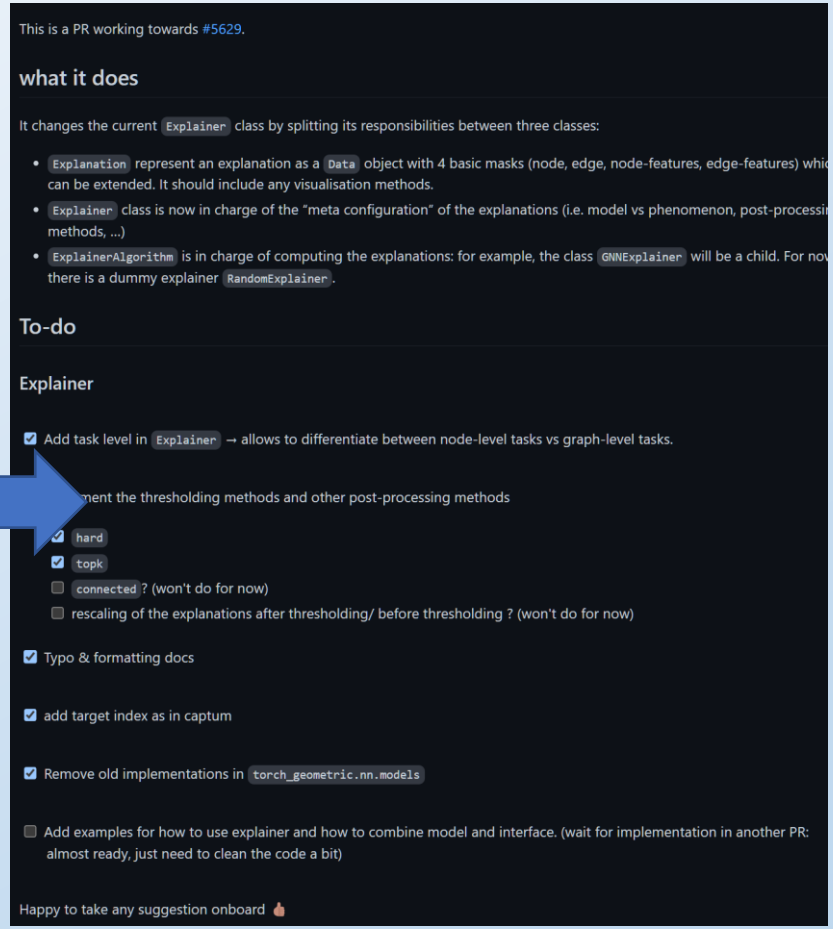
GNN explanation settings
dufourc1/pytorch_geometric

Notifications Customize

Unsubscribe

You're receiving notifications because you're watching this repository.

1 participant



This is a PR working towards #5629.

what it does

It changes the current `Explainer` class by splitting its responsibilities between three classes:

- `Explanation` represent an explanation as a `Data` object with 4 basic masks (node, edge, node-features, edge-features) whi
  can be extended. It should include any visualisation methods.
- `Explainer` class is now in charge of the "meta configuration" of the explanations (i.e. model vs phenomenon, post-processi
  methods, ...)
- `ExplainerAlgorithm` is in charge of computing the explanations: for example, the class `GNNExplainer` will be a child. For no
  there is a dummy explainer `RandomExplainer`.

To-do

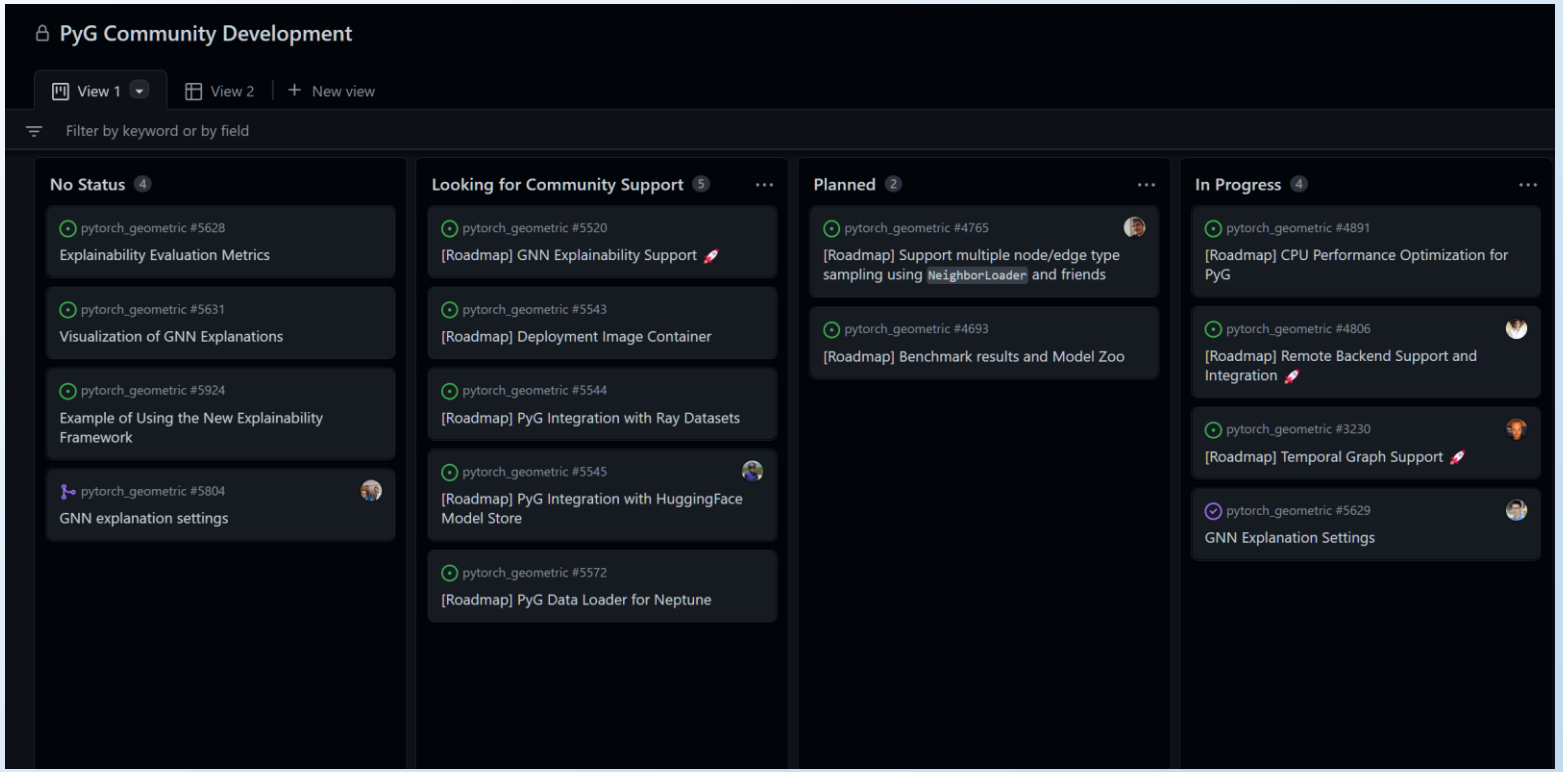
Explainer

- [x] Add task level in `Explainer` → allows to differentiate between node-level tasks vs graph-level tasks.
- [x] ment the thresholding methods and other post-processing methods
  - [x] `hard`
  - [x] `topk`
  - [ ] `connected` ? (won't do for now)
  - [ ] rescaling of the explanations after thresholding/ before thresholding ? (won't do for now)
- [x] Typo & formatting docs
- [x] add target index as in captum
- [x] Remove old implementations in `torch_geometric.nn.models`
- [ ] Add examples for how to use explainer and how to combine model and interface. (wait for implementation in another PR: almost ready, just need to clean the code a bit)

Happy to take any suggestion onboard 👍🏽

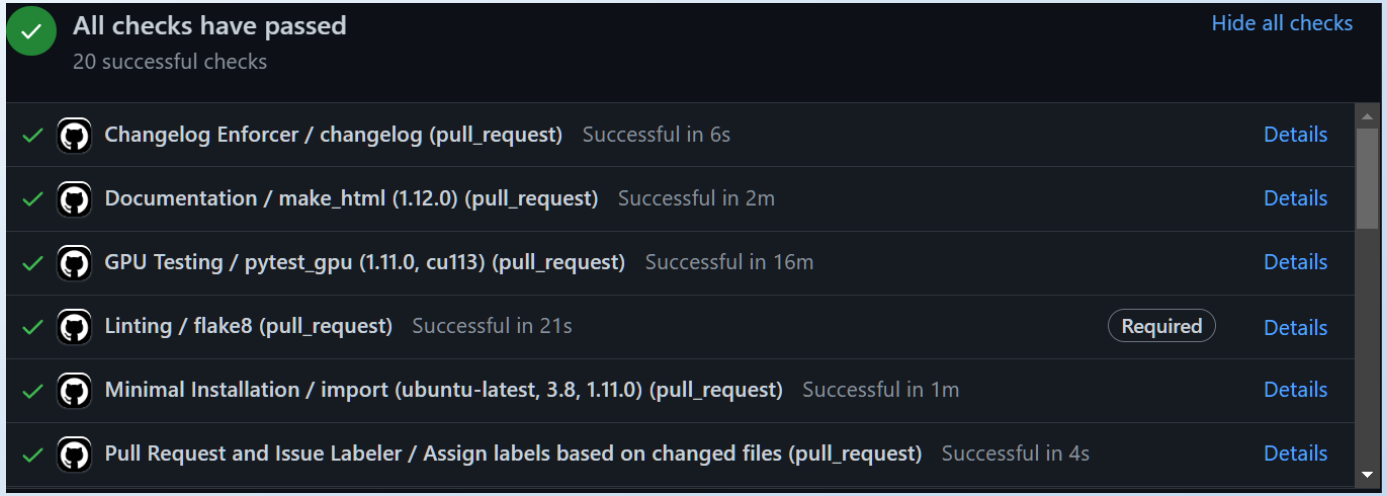
Project Tracking

- When added to projects, issues and PRs can be tracked easily



Continuous Integration

- Tests are crucial
  - Unit tests (pytest)
  - Integration tests
  - Benchmarks
- Use CI to automatically ensure that tests pass



Research Career Preparation

- Creating Impact
- Engineering Practice
- **Experience as an AP**

Teaching

- https://graph-and-geometric-learning.github.io/CPSC483-website/#/
- Having TA / head-TA experiences during PhD was valuable
  - Learn to design the pace of the course
  - Manage students and TAs
  - Understand feedbacks of the students
- Start with something you are very familiar
- Be aware of the time spent on teaching
  - Courses can be improved over the years

Lab and Students

- A **professional**, **friendly**, and **productive** environment is crucial to research and academic success
- As advisor, my role here is to help students be successful in research and future careers
  - Requires frequent communication to ensure that I'm helpful
  - Honest, direct feedback
  - Be polite
- **I ask students to not rush for papers**
  - Research impact is mostly measured by your "max", not "sum"
  - Think thoroughly on what might be an issue for readers / reviewers
  - Submission should have method and experiment sections ready **2 weeks** before the deadline

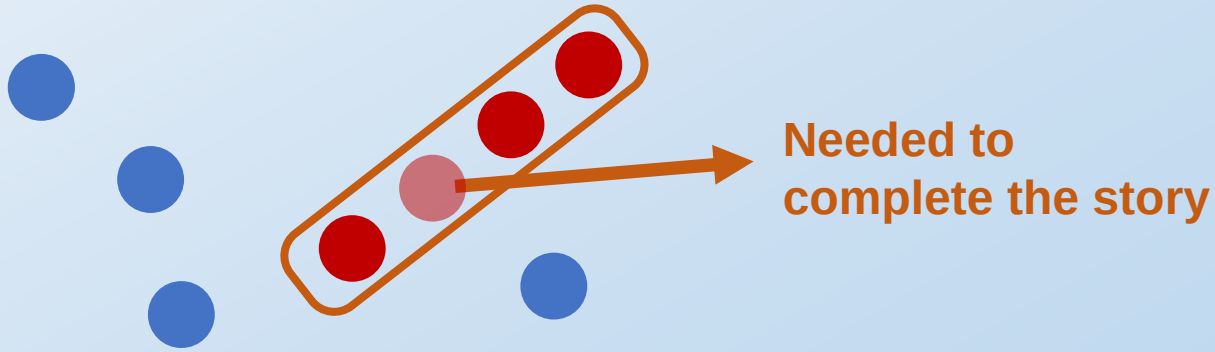
Example Timeline (1)

- Year 1: **Exploratory publications**
- Year 2: **General plan on research focus**
  - Create impact: libraries, benchmarks, surveys
  - Internship
  - Research results in your research focus
- Year 3: **Qual**
  - Major publication on your research focus
  - Collaborations and applications to demonstrate impact of your work

Have your individual plan, keeping in mind what you ultimately aim for

Example Timeline (2)

- Year 4: **Thesis proposal**
  - Connect the “dots”
  - Identify missing pieces and complete them
  - Develop research mentorship, workshop organization and other useful skills
- Year 5: **Defense (oral)**
  - Dissertation writing
  - Job applications, interviews
  - Tackle major collaboration projects



The plan can be adjusted, but it should maintain a high standard

Grants and Funding (CS Grants)

- NSF
  - Participate in review panels
  - Read example proposals (preferably within your field of research)
  - Frequently visit and read the websites in detail
  - Start thinking about potential CAREER grant topics early
- Industry grants: Google, Amazon, Meta …
  - Prior collaborations and connections would be very useful
  - Smaller-scale, but can be useful to build further collaborations
- Important non-technical aspects
  - Research integration into teaching and education outreach
  - Diversity, equity, inclusion
  - Contingency plan

Grants and Funding (Interdisciplinary)

- NIH, DoE, DoD …
  - Build prior collaborations with domain experts and show consistent effort in a particular inter-disciplinary area of research
  - Attend domain-specific workshops and conferences
  - Understand the language / jargon used in the domain

- Important non-technical aspects
  - Prior collaboration related to the interdisciplinary research
  - Collaboration plan
  - Support letters

Grants and Funding (Tips)

- Grant proposal is a great opportunity to brainstorm and systematically investigate a research idea
  - **Concrete plan**
    - **Broad statement**: “Use heterogeneous GNN to learn interactions between particles”
    - **Detailed statement**: “The message function of the GNN consists of node-specific transformations; relation-specific bilinear form; and a conservation module, where …”
    - **Compared** to previous works …
      (show that the proposed work is “transformative”)
    - Draw a **diagram** of the algorithm / model architecture
  - **Preliminary works** and evidence are important
    - Go beyond preliminary results and provide a detailed plan that builds on top of it

Thank you!!

- Creating Impact
- Engineering Practice
- Experience as an AP

- Email: rex.ying@yale.edu
- Website: https://www.cs.yale.edu/homes/ying-rex/
- Teaching: CPSC 483 (Deep Learning on Graph-structured Data) https://graph-and-geometric-learning.github.io/CPSC483-website/#/